NASA-AISRP YEAR 2 ANNUAL REPORT
Report Period: 10/1/2005-9/30/2006
Grant No: NNG05GA30G

Estimating Missing Data in Sensor Network Databases Using Data Mining to Support Space Data Analysis

Le Gruenwald
University of Oklahoma
School of Computer Science
200 Felgar Street, Room 116 EL
Norman, OK 73019
Phone: 405-623-8358
Fax: 405-325-4044
Email: ggruenwald@ou.edu

1. PROJECT ACCOMPLISHMENTS

In Year 2 of the project, we were able to complete the following tasks:

- Developed an algorithm called FARM (Freshness Association Rule Mining) to estimate missing sensor data in sensor networks, taking the temporal dimension of sensor data into consideration.
- Conducted an investigation of existing sensor data estimation techniques to compare with FARM and with another data estimation technique that we developed in Year 1, called WARM (Window Association Rule Mining).
- Investigated data and applications used in the NASA/JPL Sensor Webs project.
- Implemented FARM and existing sensor data estimation techniques and conducted experiments to compare FARM, WARM and those techniques using NASA/JPL Sensor Webs project's data and traffic data in Austin, Texas.
- Developed a data mining algorithm that deals with concept drifting when classifying data streams; implemented and compared it with the two existing data stream classification algorithms, VFDT and CVFDT.
- Developed an algorithm and data structures to mine closed frequent itemsets for data streams.
- Helped one Computer Science PhD student who is funded by this grant, Nan Jiang, complete her PhD Advisory Conference successfully.
- Helped one Computer Science Master's student who was not funded by this grant but contributed to the project by working on a Master's thesis on a related research topic, Biao Liu, complete his Master's thesis successfully.
- Published one journal paper and two conference/workshop papers, submitted one conference paper for publication, and had one journal paper under preparation.

In the following sections, we provide the details of the above tasks.

1.1. Developed an algorithm called FARM (Freshness Association Rule Mining) to estimate missing sensor data for a sensor network, taking the temporal dimension of sensor data into consideration

Our previous work in Year 1 on data streams estimation resulted in an algorithm called WARM (Association Rule Mining). WARM uses association rule mining concepts to estimate the missing sensor data readings in data streams. It identifies the sensors in the network that are related to the sensor with the missing data reading, and then uses the current readings of those related sensors to estimate the missing data reading of the missing sensor in the current round. WARM makes use of the sliding window concept, where only the last *w* rounds of data readings are stored. After receiving the first *w* rounds, every time a new round arrives, the oldest round data is discarded. In WARM, all the stored *w* rounds have the same importance in the estimation of the missing sensor data readings. In order to store the most recent data round and the relationships among sensors in the last *w* rounds, WARM uses three data structures – the Buffer, the Cube, and the Counter. To update the data structures, three algorithms were developed to estimate the missing sensor data reading – the checkBuffer, the update, and the estimateValue.

However, WARM has a number of deficiencies. First WARM has two relatively complicated data structures, the Cube and the Counter, that consume extra memory space and, at the same time, result in more time to estimate because of the need to update them after every new round of sensor data readings arrives. Second, it is possible in WARM to have more than one estimated value for one missing value where we end up choosing an estimated value randomly, which is not desirable. Third, freshness of data is not considered appropriately in WARM. By freshness of data, we mean that new data is more important than old data as in weather applications. In WARM, only the last *w* rounds in the sliding window are considered in estimation, but the effect of the sliding window size on estimation is not taken into account. In addition, all the rounds stored in the sliding window have the same importance; this means that if the sliding window size is big, the current round has the same estimation effect as the last round in the sliding window, which is not appropriate for many real-time applications such as climate applications. Fourth, WARM can only answer queries related to the data stored in the sliding window. Since we should scan the data in data streams only once, WARM stores only the last *w* rounds and disregards all other rounds. For that, there is no way to respond to user queries related to the past data.

In Year 2, we developed a data estimation approach, called FARM (Freshness Association Rule Mining) that aims to eliminate WARM's deficiencies. Like WARM, FARM also uses association rule mining to identify the sensors that are related to the sensor with a missing value and uses the data readings of those sensors to estimate the missing value in the current round. FARM takes freshness of data into consideration, which is crucial in many real-time applications where recent data is more important than old data in the estimation of the missing value that occurs in the most recent round of data readings. To consider freshness of data, all rounds of data readings in FARM participate in the estimation process, where each round has a different level of contribution (called weight) to the estimation value. The weight of a round (x) is *p* times of the weight of the previous round (x-1), where *p* is a user-defined value greater than 1, called a damped factor. This factor represents the importance of a round of data readings compared with the previous one in the estimation process. The order x reflects the recency of the data in that

round. The more recent the data are, the higher weight they have, and subsequently, the more they will contribute to the calculation of the estimated values for the missing data. In order to estimate, FARM determines the sensors that are related to the sensor with the missing value so that they can participate in the estimation of the missing sensor data reading. To store the most recent data round and the relationships between every pair of sensors, FARM uses the Buffer and the 2D Ragged Array of Objects. To estimate the missing data, FARM employs three algorithms – checkBuffer, update, and estimateValue – as in WARM but with appropriate modifications to reflect the changes in data mining and data structures.

An added advantage of FARM is that by using the weights of the states of the diagonal entries in the two dimensional ragged array of objects, the Retrieve Data algorithm in FARM is able to retrieve the whole data again at any point in time, and thus, can respond to all user queries. This eliminates the problem associated with data stream applications where no limited memory can hold the whole data streams because memory is limited and data streams arrive continuously. That problem exists in WARM: there is no way to access the whole data again; i.e., WARM can not answer queries related to the data that is not stored in the window.

1.2. Conducted an investigation of existing sensor data estimation techniques to compare with FARM and with another data estimation technique that we developed in Year 1, called WARM (Window Association Rule Mining)

In Year 1, we identified a number of popular statistical methods for data estimation existing in the literature [Allison 2002; Dempster, 1977; Gelman, 1995; Iannacchione, 1982; McLachlan, 1997; Rubin, 1987; Rubin, 1996; Shafer, 1995], such as Mean Substitution, Simple Linear Regression, Cold Deck Imputation, Hot Deck Imputation, and Expectation Maximization, Maximum Likelihood, and Multiple Imputations. In Year 2, we investigated further into the following existing techniques that handle missing sensor/stream data. In the NASA/JPL Sensor Webs project [NASA, 2006], if one sensor fails, its neighboring sensors compensate for the last data by increasing their sampling rates. There are several disadvantages with this approach. First, there must be a tight collaboration among sensors for a sensor to know that its neighboring sensor has failed. This increases power consumption on every sensor even during its normal operation and, thus, is impractical for sensor networks where data missing/corruption is expected not to be a rare occurrence. Second, the approach does not address how sampling rates of neighboring sensors should be adjusted in order to guarantee a good quality of service in terms of data accuracy, query real-time constraints as well as sensors' power consumption. And third, if several sensors that are next to each other, say S2, S3, and S4, fail at the same time, which is possible in some harsh environments like the Mars planet, adjusting the sampling rates of S1 and S5 would not compensate for the data lost caused by S3, assuming that S1 is next to S2 and S5 is next to S4.

Some probabilistic models, such the BBQ system [Deshpande, 2005], are designed to predict missing values in data streams. The BBQ system is learned from a set of training data which consist of the readings of all monitored attributes of data streams over a period of time. During the training period, the system computes the initial correlations between the monitored attributes in order to build its basic model. After training, the system adjusts the initial correlations every time new values for the monitored attributes are observed. The BBQ system has two

disadvantages. First, it may take a lot of time to build the basic model during the training period if some missing values exist in the training data. Second, it cannot answer user queries before building the basic model; i.e., estimation of missing values cannot be performed before the basic model is constructed.

SPIRIT [Papadimitriou, 2005] uses auto-regression as its basic forecasting model to estimate missing values in data streams. SPIRIT spots correlations on numerical streams and extracts the hidden variables that summarize the key trends in the entire stream collection. To estimate the missing values, SPIRIT applies auto-regression on the extracted hidden variables and uses the results to predict the values of the missing data. So, whenever there is a missing value in any round of data, SPIRIT uses the forecast based on the values in the previous round to estimate the missing values in the current round. One of the disadvantages with estimation using SPIRIT is related to the accuracy of estimation. It is true that SPIRIT estimates very fast and consumes just a little memory space, but its accuracy is not as good as other estimation techniques.

TinyDB [Madden, 2005] is a query processing system for extracting information from a network of special type of sensors. Given a query specifying the user data interests, TinyDB collects that data from motes in the environment, filters it, aggregates it together, and routes it out to a PC. TinyDB does this via power-efficient in-network processing algorithms. TinyDB estimates the missing values by taking the average of all the values reported by the other sensors in the current round. This approach has many disadvantages: First, the accuracy of this approach is in doubt because only the readings of other sensors at the current round are used to estimate the missing sensor value; it may be that the missing sensor is not related at all to those other sensors. Second, this approach will have bad accuracy results especially if there is multiple sensors failure at the same time; and more specifically, if related sensors miss their value at the same time.

1.3. Investigated data and applications used in the NASA/JPL Sensor Webs project

Examining data and applications used by the NASA/JPL Sensor Webs project [NASA, 2006], we were able to identify a dataset that we can use to test our algorithms for sensor data applications in which data freshness is important. The dataset consists of 13 sensors where all sensors also report their data readings to a single server every *5 minutes*. The sensors are embedded in the Huntington Botanical Gardens in San Marino, California. The sensors report the air temperature of several places in the gardens for different time intervals. The actual sensors' readings are used as an input data to our simulation. Using this dataset, we want to study the effect of freshness of data since this is a climate dataset and the importance of data recency is reflected here. For example, in this dataset, there is a great possibility that the air temperature reported by a certain sensor at 6:00 pm, for instance, is the same as the air temperature reported by the same sensor at 6:05 pm.

1.4. Implemented FARM and existing data estimation techniques and conducted experiments to compare FARM, WARM and those techniques using NASA/JPL Sensor Webs Project's dataset and Austin's Traffic Dataset

We wrote Java programs to implement FARM and existing data estimation techniques. We then conducted simulation experiments to compare the performances of FARM and WARM with SPIRIT, TinyDB, and four statistical estimation methods: (1) The Simple Linear Regression (SLR) approach: using estimated value based on the readings stored in the data model of this particular sensor and its next neighbor, (2) The Multiple Linear Regression (MLR) approach: using estimated value based on the readings stored in the data model of this particular sensor and its next 10 neighbors, (3) The Curve Regression (CE) approach: using estimated value based on the readings stored in the data model of this particular sensor and its next neighbor, and (4) The Average estimation approach: using the average of all the past readings for the sensor with the missing value in the current round as an estimate. The performance metrics include estimation accuracy, estimation inability that represents the percentage of cases that cannot be estimated by the estimateValue algorithm alone, execution time, and storage space. We used both the NASA/JPL Sensor Webs Project's dataset as well as the traffic data collected by the Department of Transportation in Austin, Texas [Austin, 2005]. The results, as demonstrated through Tables 1 – 6, can be summarized as follows. For applications where recent data is more important than old data, FARM yields the best accuracy. For applications where old data is as important as recent data, WARM yields the best accuracy. SLR has the best execution time and SPIRIT consumes the least storage space. The statistical approaches can estimate very fast and consume little memory space but they all fail to satisfy the accurate estimation property. Therefore, the statistical approaches do not fit most applications where accuracy cannot be sacrificed. SPIRIT also estimates very fast and consumes the least memory space, but it fails to satisfy the accurate estimation property for applications where freshness of data is considered. Both FARM and WARM have feasible storage space and acceptable estimation speed for all applications. However, their accuracy differs according to the types of applications. For applications where recent data is more important than old data, FARM gives the overall best results. For applications where old data is as important as recent data, WARM should be used.

Table 1. Estimation Error for NASA data set

	Estimation Error for NASA data	How many % the best approach is better?
FARM	**0.00487**	**Best Approach**
WARM	0.00636	23.43%
TinyDB	0.0085	42.71%
SPIRIT	0.0116	58.02%
Average	0.015	67.53%
MLR	0.121	95.98%
SLR	0.342	98.58%
CE	0.346	98.59%

Table 2. Estimation Error for traffic data set

	Estimation Error for traffic data	How many % the best approach is better?
WARM	**0.090**	**Best Approach**
SPIRIT	0.0932	3.43%
FARM	0.134	32.84%
TinyDB	0.137	34.31%
Average	0.1445	37.72%
MLR	0.493	81.74%
CE	2.253	96.01%
SLR	2.286	96.06

Table 3. Execution time for NASA data

	Execution time for NASA data (seconds)	How many % the best approach is better?
SLR	**0.044**	**Best Approach**
CE	0.049	10.2%
MLR	0.059	25.42%
SPIRIT	0.161	72.67%
Average	0.200	78%
TinyDB	0.210	79.04%
FARM	0.261	83.14%
WARM	0.301	85.38%

Table 4. Execution time for traffic data

	Execution time for traffic data (seconds)	How many % the best approach is better?
SLR	**0.053**	**Best Approach**
CE	0.061	13.11%
MLR	0.085	37.65%
SPIRIT	0.270	80.37%
TinyDB	0.309	82.85%
Average	0.449	88.2%
FARM	0.621	91.47%
WARM	3.145	98.31%

Table 5. Storage Space for NASA data

	Storage Space for NASA data (KB)	How many % the best approach is better?
SPIRIT	0.1	**Best Approach**
Average, TinyDB	0.15	33.33%
SLR	1.32	92.42%
CE	1.32	92.42%
MLR	5.22	98.08%
FARM	43.1	99.77%
WARM	119.21	99.92%

Table 6. Storage Space for traffic data

	Storage Space for traffic data (KB)	How many % the best approach is better?
SPIRIT	0.85	**Best Approach**
Average, TinyDB	1.27	33.07%
SLR	3.12	72.76%
CE	3.12	72.76%
MLR	5.47	84.47%
FARM	911	99.91%
WARM	5013	99.98%

1.5. Developed a data mining algorithm that deals with concept drifting when classifying data streams; implemented and compared it with the two existing data stream classification algorithms, VFDT and CVFDT

Many recent applications like sensor networks and network traffic monitoring system generate data streams that continuously arrive in high speed. The logic of data generation, or the concept that we try to learn from the data, might be constantly changing over time. This is known as concept drifting. Existing techniques for classifying data streams assume that either no concept drifting happens, or concept drifting frequently occurs, and thus, spend a large amount of system resources to monitor their target models for changes. However, both assumptions are not necessarily true in real-word applications. As a result, existing techniques either cannot handle concept drifting if it does occur, or waste tremendous precious system resources if concept remains stationary. Addressing this issue, in Year 2, we introduced a concept-drifting indicator and developed a new algorithm, called CSDI. Instead of monitoring the entire target model, which is expensive and impractical, we only monitor this indicator, and check the target model only when the indicator reveals concept drifting. Because tracking the concept-drifting indicator only needs a very small amount of system resources, our algorithm can remain efficient regardless of whether the concept is drifting or stationary.

CSDI works as follows. It first uses the existing algorithm, called VFDT[ ], to construct an initial decision tree, and then uses this initial decision tree to classify the succeeding chunk of sample records to obtain the normal concept-drifting indicator. The concept-drifting indicator is the error rate that measures the accuracy of the constructed decision tree in classifying records, and the concept-drifting indicator that is obtained initially when no concept drifting has occurred is called the normal concept-drifting indicator or normal error rate. Based on the obtained normal concept-drifting indicator and a user-defined indicator factor, CSDI then computes the concept-drifting threshold that is the upper bound of the normal concept-drifting indicator. Concept drifting is believed to have occurred if the concept-drifting indicator reaches the threshold. After initialization, CSDI starts to classify records by using the initial decision tree and to count the number of records that were classified incorrectly on each node. CSDI periodically (for each checking interval) checks the concept-drifting indicator to detect concept drifting. If concept drifting is detected, CSDI will start the searching process to locate all the invalid nodes under the new concept, and then update the tree.

We conducted experiments comparing CSDI with two existing algorithms, VFDT [Domingos, 2000] and CVFDT [Hulten, 2001], using both synthetic and real data. We obtained the

following results. If no concept drifting happens, VFDT is the best among the compared algorithms. But if there is concept drifting, the accuracy of VFDT decreases drastically and cannot be accepted. Although CVFDT can handle concept drifting, it uses too much system resource no matter whether or not there is concept drifting or how often concept drifting occurs. Our algorithm, CSDI, uses a little more system resource but achieves more accuracy and much more efficiency than CVFDT does. Therefore, CSDI can remain efficient and accurate regardless of whether there is concept drifting or not. Also CSDI can scale up very well when the number of attributes in data streams increases. Figure 1 shows the performance comparison in terms of classification error rate when the number of times that concept drifting occurs is varied.

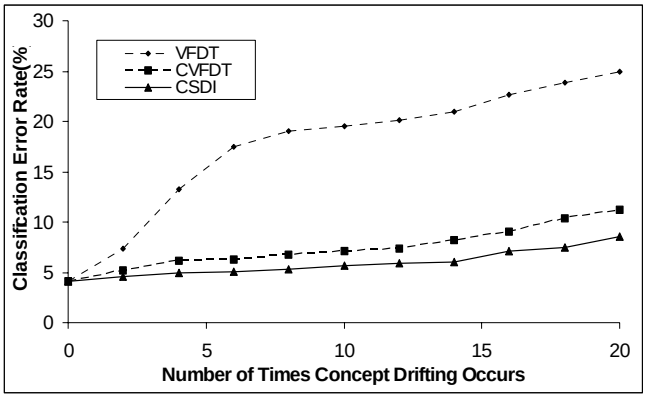

Figure 1: The effect of number of times concept drifting occurs on the classification error rate

In Year 3, we plan to incorporate this algorithm into FARM to improve data estimation.

1.6. Developed an algorithm and data structures to mine closed frequent itemsets for data streams

Our data estimation techniques, both WARM and FARM, rely on the ability of our modified Apriori algorithm to discover association rules among sensors. However, the modified algorithm is based on frequent itemsets, and thus consumes a lot of time and space. To reduce those overheads, we use the concept of frequent closed itemsets instead. Mining frequent closed itemsets provides complete and condensed information for non-redundant association rules generation. Extensive studies have been done on mining frequent closed itemsets, but they are mainly intended for traditional transaction databases, and thus do not take data stream characteristics into consideration. In Year 2, we developed a novel approach for mining closed frequent itemsets over data streams, called CFI-Stream. It computes and maintains closed itemsets online and incrementally, and can output the current closed frequent itemsets in real time based on users' specified thresholds. When a transaction arrives or leaves the current data stream sliding window, the algorithm checks each itemset in the transaction on the fly and updates the associated closed itemsets' supports. Current closed itemsets are maintained and updated in real time in a lexicographical ordered tree data structure, called DIU (Direct Update) tree. Each node in the DIU tree represents a closed itemset. There are k levels in the DIU tree;

each level i stores the closed i-itemsets. The parameter k is the maximum length of the current closed itemsets. Each node in the DIU tree stores a closed itemset, its current support information, and the links to its immediate parent and children nodes.

Different from previous closure checking techniques which require multiple scans over data [Pei, 2000; Chang, 2003; Xu, 2004; Lin, 2005], our method performs the closure checking on the fly with only one scan over data streams. It updates only the supports of the associated closed itemsets in the DIU tree online, which reduces the computation time and provides real time updated results. Our algorithm is an incremental algorithm where we check for closed itemsets and update their associated supports based on the previous mining results. This is more efficient as compared with mining approaches that rescan and regenerate all closed itemsets when a new transaction arrives.

Compared with other data stream mining techniques [Manku, 2002; Chi, 2004; Lin, 2005] etc., we only store the information of current closed itemsets in the DIU tree, which is a compact and complete representation of all itemsets and their support information. The current closed frequent itemsets can be output in real time based on users' specified thresholds by browsing the DIU tree. Also, our algorithm handles the concept-drifting problem in data streams by storing all current closed itemsets in the DIU tree from which all itemsets and their support information can be incrementally updated.

We compared our algorithm with Moment [Chi, 2004], which is the current state-of-the-art algorithm to mine closed itemsets in data streams. For the performance study, two synthetic datasets T10.I6.D100K and T5.I4.D100K-AB were used. Experimental results show that our method can achieve better performance than Moment in terms of both time and space efficiency, especially when the minimum support is low and the datasets is dense. Figure 2 illustrates the runtime performance comparison of the two algorithms for one of the datasets.

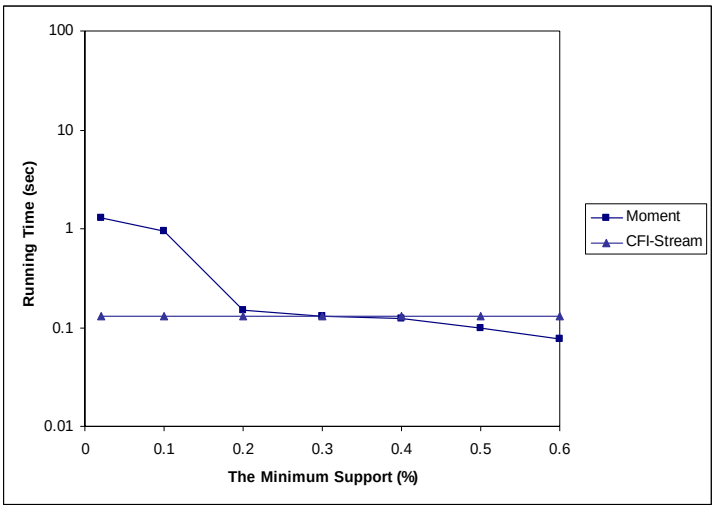

Figure 2. Runtime of CFI-Stream and Moment using the T10.I6.D100K data set

2. CHALLENGES

The major challenge that we faced in Year 2 was how to obtain appropriate datasets for our performance studies. We contacted the NASA/JPL research staff and carefully studied their Sensor Webs project in detail. We finally were able to obtain an appropriate dataset that we could use. But to evaluate the scalability of our algorithms, we would need larger datasets than the one we had. In addition, we also would need to investigate queries on sensor data and implement those queries in our simulation models in order to study the effects of data estimation algorithms on response time of real-life queries.

3. PLANS FOR NEXT YEAR

In Year 3, we plan to work on the following tasks:

- Revise FARM to include spatial data mining.
- Work with NASA scientists to incorporate NASA data semantics into FARM.
- Conduct a theoretical analysis of FARM.
- Collect additional NASA data and queries aand conduct further performance evaluations.
- Extend FARM to include distributed sensor networks and mobile sensor networks.
- Conduct theoretical analysis and experiments comparing FARM with existing data estimation techniques for distributed sensor networks and mobile sensor networks using NASA data and synthetic data.
- Incorporate classification, clustering, and frequent closed items association rule mining into FARM to improve data estimation.

4. PUBLICATIONS AND PRESENTATIONS

- Nan Jiang and Le Gruenwald, "Research Issues in Association Rule Mining for Data Streams," SIGMOD RECORD, Vol. 35, No. 1, March 2006.
- Biao Liu, "Classify Data Streams using Concept-Drifting Detection Indicator," Master's Thesis, School of Computer Science, University of Oklahoma, May 2006.
- Nan Jiang and Le Gruenwald, "CFI – Stream: Mining Closed Frequent Itemsets in Data Streams," accepted for publication in the proceedings of the ACM International Conference on Knowledge and Data Discovery (KDD), August 2006.
- Nan Jiang and Le Gruenwald, "An Efficient Algorithm to Mine Online Data Streams," accepted for publication in the proceedings of the International Workshop on Temporal Data Mining, August 2006.
- Mazen Aboukhamis and Le Gruenwald, "Using Data Mining to Estimate Missing Sensor Data," submitted to IEEE International Conference on Data Engineering, July 2006
- Biao Liu, Le Gruenwald, Nan Jiang and Mazen Aboukhamis, "A Survey of Stream Data Mining Techniques", under preparation, to be submitted to Data and Knowledge Engineering, 2006.

5. REFERENCES

[Agrawal, 1993] Rakesh Agrawal, Tomasz Imielinski, Arun Swami. “Mining Association Rules between Sets of Items in Large Databases”, the ACM SIGMOD International Conference on Management of Data, pp. 207-216, May 1993.

[Austin, 2005] Austin Freeway ITS Data Archive, http://asutindata.tamu.edu/default.asp, accessed December 2005

[Chang, 2003] Joong Hyuk Chang, Won Suk Lee, Aoying Zhou. “Finding Recent Frequent Itemsets Adaptively over Online Data Streams.” ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining; August 2003.

[Chi, 2004] Yun Chi, Haixun Wang, Philip S. Yu, Richard R. “ Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window.” IEEE Int'l Conf. on Data Mining; November 2004.

[Dempster, 1977] A. Dempster, N. Laird, and D. Rubin. “Maximum Likelihood from Incomplete Data via the EM Algorithm”, Journal of the Royal Statistical Society, Series B, 39(1), pages 1-38, 1977.

[Deshpande, 2005] A. Deshpande, S. Madden, C. Guestrin. “Using Probabilistic Models for Data Management in Acquisitional Environments.” CIDR, 2005.

[Domingos, 2000] P. Domingos and G. Hulten. “Mining high-speed data streams.” ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000, P71-80.

[Gelman, 1995] A. Gelman, J. Carlin, H. Stern, and D. Rubin. “Bayesian Data Analysis. "Chapman & Hall, 1995.

[Hulten, 2001] G. Hulten, L. Spencer, P. Domingos. “Mining Time-Changing Data Streams.” ACM International Conference on Knowledge Discovery and Data Mining, 2001, P97-106

[Lin, 2005] Chih-Hsiang Lin, Ding-Ying Chiu, Yi-Hung Wu, Arbee L. P. Chen. “Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window.” SIAM International Conference on Data Mining; April 2005.

[Madden, 2005] S. Madden, M. Franklin, J. Hellerstein, W. Hong. “TinyDB: An Acquisitional Query Processing System for Sensor Networks.” TODS, 2005.

[Manku, 2002] Gurmeet Singh Manku, Rajeev Motwani. “Approximate Frequency Counts over Data Streams.” International Conference on Very Large Data Bases, 2002.

[McLachlan, 1997] G. McLachlan and K. Thriyambakam. “The EM Algorithm and Extensions”. New York: John Wiley & Sons, 1997.

[NASA, 2006] NASA/JPL Sensor Webs Project, http://caupanga.huntington.org/swim/, accessed Jan 2006.

[Papadimitriou, 2005] S. Papadimitriou, J. Sun, C. Faloutsos. "Pattern Discovery in Multiple Time-Series**."** International Conference on Very Large Data Bases, 2005, pp .697-708.

[Pei, 2000] J. Pei, J. Han, and R. Mao. "Closet: An efficient algorithm for mining frequent closed itemsets." SIGMOD International Workshop on Data Mining and Knowledge Discovery, May 2000.

[Rubin, 1987] D. Rubin, "Multiple Imputations for Nonresponce in Surveys". New York: John Wiley & Sons, 1987.

[Rubin, 1996] D. Rubin. "Multiple Imputations after 18 Years", Journal of the American Statistical Association, 91, pp. 473-478, 1996.

[Shafer, 1995] J. Shafer. "Model-Based Imputations of Census Short-Form Items", The Annual Research Conference, Washington, DC: Bureau of the Census, pages 267-299, 1995.

[Xu, 2004] Jeffrey Xu Yu, Zhihong Chong, Hongjun Lu, Aoying Zhou. "False Positive or False Negative: Mining Frequent Itemsets from High Speed Transactional Data Streams." International Conference on Very Large Data Bases, 2004.

[Yi, 2000] B.-K. Yi, N. Sidiropoulos, T. Johnson, H. V. Jagadish, C. Faloutsos, A. Biliris. On-Line Data Mining for Co-Evolving Time Sequences, International Conference on Data Engineering, 2000, pp. 13-22.

[Wang, 2003] Wang H., MiningConcept-Drifting Data Streams using Ensemble Classifiers. In 9th ACM International Conference on Knowledge Discovery and Data Mining SIGKDD, August 2003.